CMM®–Compliant XP

Abstract

Extreme Programming (XP) and the other “Agile Methods” have recently burst onto the scene, and many people are talking and asking lots of questions. Those questions take many forms, but in some organizations, they include, “Should we do XP? Or CMM[]?” This question seems natural, since XP and CMM appear to be incompatible.

- Light vs. Rigorous
- People-oriented vs. process-centric
- Change-embracing vs. change-controlling
- Barely-adequate process vs. Repeatable, Defined, Managed and Optimizing process
- 12 practices vs. 368 Key Practices

But incompatibility need not be a foregone conclusion. The proponents of XP readily acknowledge that it is not appropriate for every project. In fact, they recommend XP for a very specific subset of projects, those that are small and exploratory in nature. At the same time, the CMM recognizes that different projects may require different processes. An Organization’s Standard Software Process (OSSP) may, in fact be a suite of options, in which case, establishing a Project’s Defined Software Process would involve choosing from among those options. (Refer to the Level 3 KPAs OPD and ISM for more information.)

Interestingly, the types of projects for which XP was designed, are precisely the ones with which organizations engaged in CMM-based process improvement programs have the most trouble. These “odd” projects are too small, too fast moving, and the results are too ill defined for the “normal” processes. But at the same time, we recognize that careful management is as important with these innovative projects as with any other.

So, could XP be one of the available options in your OSSP? No, and Yes. XP, as defined “out of the box” does not satisfy a number of the Goals of the CMM. However, with a few minor extensions, it can be made goal-compliant without sacrificing most of its “agility”.

Introduction

The author of XP, Kent Beck is explicit that Extreme Programming is designed for use in exploratory projects. That is, it is best suited for projects where the end results are only vaguely known at the start of the project, or there are technological or user-interface questions that the project is supposed to answer. It is also aimed at projects that are characterized by a small co-located development team of up to 10 people. (For more information on XP, please refer to my article, “XP—An Overview” which can be found at http://www.ASKProcess.com/Articles.)

The types of projects for which XP was designed also happen to be those that have the most trouble implementing disciplined CMM-based software processes. Therefore, there may be an opportunity to capitalize on the work of Kent Beck, to craft a version of XP that would fit into a CMM organizational environment. To that end, this article answers the question:

Can XP be used as a CMM-compliant option for exploratory projects?

[] “CMM” and “Capability Maturity Model” are registered in the US Patent and Trademark Office.

In answering this question, we will look specifically at each level of the Software CMM and discuss compliance with the Key Process Areas (KPAs). We will point out where standard XP practices are likely to comply with CMM Goals, and we will discuss extensions that may be necessary to achieve compliance with other Goals. The details of complying with each KPA, including analyzing each Key Practice is beyond the scope of this article. But the author would welcome the opportunity to engage in that level of discussion with any interested correspondent.

Level 2

Requirements Management

Because an XP project is exploratory in nature, the Requirements are defined only at a very high level. A collaborative effort between the customer and the developers results in the Requirements being documented in two ways:

- Metaphor – This is the over-all concept of what the final system will look like. This is the goal of the project, defined in the customer's terms.
- Stories – These are the descriptions of the features of the system. Each feature is described in a customer-oriented story that fits on a 3x5 card.

XP's short "release" iterations are the key to maintaining consistency between the requirements and other work products. Each release ends with the delivery of interesting or useful features to the customer. Then, in the "Planning Game" that marks the beginning of the next iteration, the developers and the customer:

- Update the Metaphor and Stories on the basis of what both have learned in prior iterations and
- Select the features to be implemented and updated in the next iteration.

Because these iterations are designed to be quite short (a few weeks), this realignment is done on a regular basis.

The Goals of Requirements Management may be satisfied by the standard XP practices.

Software Project Planning

XP calls for the project schedule to be estimated at the beginning of the project, and then to be adjusted during the "Planning Game" at the beginning of each iteration, as a result of what has been learned to date. Although XP does not consider the concept of estimating size and the other items required by the CMM, we can easily add those estimates as steps on the way to the schedule estimate.

Because of the nebulous nature of the project requirements or the technical issues being explored in an XP project, detailed size estimation by component or module is not a reasonable expectation. However, the project maintains a discrete list of features that can be the basis for the estimates.

- The **Size** of each feature can use "measures" such as "Small", "Medium", "Large" and "Easy", "Moderate", "Complex". Any feature could then have a "size" of "Small & Complex" or "Large & Moderate".
- An estimate of the **Effort** required for each of the nine Size measures described above can be made based on any available historical data and the team's judgment.

(Naturally, as the project progresses, these estimates can be refined using actual data from the project.) So the Effort required for a feature would be derived directly from its Size.

- As with most projects, **Cost** is usually the payroll cost of the computed effort. However, there may be times when a particular feature may have additional costs associated with it, such as special equipment.
- **Schedule** can easily be estimated (both for the entire project and for the next iteration) based on the total effort required for the features to be implemented in that iteration and the available effort of the project team. Naturally, dependencies must be identified and accounted for.
- **Critical Computer Resources** can be "guessed at" for the project as a whole. But in exploratory projects, CCR's are often unknown, and the subject of the exploration. Therefore, these estimates are likely to be inaccurate.

Communicating commitment between the customer and the developers is a hallmark of XP. At the same time, because XP is intended to be used in exploratory projects, project commitments are recognized to be tenuous and subject to revision as the project moves forward.

In the Planning Game that marks the beginning of each iteration, the customer and the developers consolidate what they have learned, adjust their expectations and plans, and choose the features to be implementing in that iteration. In this way, the project plans and commitments are discussed, adjusted and documented at the beginning of each iteration. Although the amount of detail in the documented plans and commitments is scant in an XP project as compared to most CMM compliant projects, it is good enough and appropriate for use in such short iterations.

The Goals of Software Project Planning can be satisfied by the XP practices with some extensions.

Software Project Tracking and Oversight

The project's actual performance is compared to the plan during the "Planning Game" for each iteration. And because each iteration is only a few weeks in length, this tracking is done on a regular basis.

The "Planning Game" is "played" by all of the developers and the customers (or customer representatives). Because XP project teams are small and co-located, this covers almost all of the affected groups and individuals. The only people who might be left out would be supporting functions like SQA and SCM, and possibly other engineering groups (e.g. Hardware) if the project includes such things. Adding appropriate representatives to the Planning Game meetings can correct these deficiencies.

Because of the exploratory nature of an XP project, XP does not include a mechanism for recovering when the actual performance differs from the plan. Rather, it assumes that the plan must change to match "reality". Therefore, the "Planning Game" includes adjusting the project's plans for each iteration.

XP does not address the question of reporting status up the management chain, but such reporting could easily be made an outcome of each "Planning Game".

The Goals of Software Project Tracking and Oversight can be satisfied by the XP practices with some minor extensions.

Software Subcontract Management

XP is designed for small cohesive co-located teams. Therefore, it would not be appropriate for a project that includes subcontractors.

Software Subcontract Management is not applicable in an XP project.

Software Quality Assurance

SQA is an organizational process. Assuming that the XP processes are CMM compliant, the organizational SQA function will apply to XP projects just as to any other.

SQA is fully compatible with an XP project.

Software Configuration Management

XP does not address SCM at all. However, many of the practices of XP clearly require solid SCM practices for a smooth-running project.

- XP calls for many small "releases", one every few weeks, with each release serving as the baseline for the following ones. This concept infers strong baselining mechanisms and careful version control of the code and other components of each release.
- "Refactoring" is XP's mechanism for doing incremental redesign of the system as the developers and the customer learn throughout the project. Implementing these re-designs would require strong code control.
- Collective Ownership of the code would require careful version control and continual availability of all components of the system.

When viewed through the lens of SCM, the Planning Game for each iteration *is* the project's regular CCB meeting.

- It includes all of the affected groups and individuals.
- The result of the just-completed iteration is accepted as the current product baseline.
- Changes to the Requirements are discussed, approved and scheduled for implementation, or disapproved.
- The release plan (what features will be implemented in each future iteration) is discussed and changed as needed.
- The exact content and delivery date for the next iteration is agreed upon.

The only part of SCM that is missing from XP is the project's SCM plan. XP appears to *assume* that the SCM activities are planned, but for CMM compliance, an explicit SCM plan would have to be documented. Given the importance of SCM to an XP Project, this would be a prudent addition.

The Goals of Software Configuration Management can be satisfied by the XP practices with some minor extensions.

Level 3

Organizational Process Focus and *Organizational Process Definition*

OPF and OPD are organizational processes, and not related to XP.

Organizational Process Focus and Organizational Process Definition are not applicable in an XP project.

Training Program

TP is an organizational process. XP assumes appropriate training for all developers.

Training Program is fully compatible with an XP project.

Integrated Software Management

XP is based on careful compliance with the XP practices. Assuming that the organization's XP process is implemented in a way that assures it is CMM compliant…

The Goals of Integrated Software Management can be satisfied by appropriately tailored XP practices.

Software Product Engineering

The major focus of XP is on the software engineering tasks. The XP practices explicitly define how requirements, architecture, design, coding and testing are to be accomplished. The XP engineering practices are well defined and tightly integrated, and XP specifies that those practices **must** be performed as defined in order to achieve positive results.

The "Planning Game" that is undertaken at the beginning of each XP iteration includes re-visiting and updating the requirements for the system based on what was learned in prior iterations. Obviously, these updates may result in prior design or coding work being invalidated. XP *assumes* that this will happen in exploratory projects, and so the XP practices are designed to assure consistency in the environment of changing expectations.

XP includes a practice called "refactoring". This is essentially incremental re-design of already implemented parts of the system in order to correct for new understanding that was generated in prior increments. Refoactoring is not massive re-design and re-implementation; rather, it is a method for applying needed re-design to the parts of the system that will be touched in the up-coming iteration.

Naturally, code that is invalidated by new understanding will be corrected, either by removing it with the intention of replacing it in a later iteration, or by reworking it as a part of the up-coming iteration. Either way, both the design and the code is always kept consistent with the developers' and the customers' evolving understanding of the requirements.

The Goals of Software Product Engineering may be satisfied by the standard XP practices.

Intergroup Coordination

XP is designed for use by a small, co-located team that includes all of the developers as well as the customer. As such, the only other "affected groups and individuals" might be those that who provide services to the project (e.g. SQA and possibly SCM), and possibly other engineering groups (e.g. Hardware) if the project includes such things. Therefore, all of the IC goals can be satisfied by including such people in the initial planning and the "Planning Game" that marks the beginning of each iteration of the project.

The Goals of Intergroup Coordination can be satisfied by the XP practices with some minor extensions.

Peer Reviews

The most unique feature of XP is "Pair Programming". This practice dictates that *all* activities (design, coding and testing) are done by two people working together – one at the computer keyboard, and the other observing in real time. XP advocates believe this to be the most effective type of peer review possible. (It certainly optimizes the principle of removing defects as soon as possible after they are inserted!)

Because the CMM does not put limits on the types of activities that can be used to satisfy the goals of this KPA, I believe that Pair Programming can satisfy the PR Goals.

The Goals of Peer Reviews may be satisfied by the standard XP practices.

Level 4 and Level 5

Quantitative Process Management, ***Software Quality Management*** and ***Defect Prevention***

Although XP is strongly quality-focused, it does not include the concept of quantitative management of either quality or process performance. Satisfying the goals of these KPAs would require the addition of appropriate goal-setting and measurement activities.

The Goals of Quantitative Process Management, Software Quality Management and Defect prevention would require additional activities beyond the standard XP practices.

Technology Change Management and ***Process Change Management***

TCM and PCM are organizational processes, and are not directly related to the activities of a specific project.

Technology Change Management and Process Change Management are not applicable in an XP project.

Summary

Based on this level of analysis, it appears that XP *can* be used as an option for certain kinds of projects in a CMM-compliant environment. In order to maintain CMM compliance at Level 2 or 3, some extensions to XP's standard practices would be required. But those extensions appear to be minor, and should not have a significant impact on the Agility of the XP method. Compliance at CMM Level 4 or 5 would require more significant additions to the XP practices. But, the organizational learning that would be gained from XP projects in a Level 4 or Level 5 organization would more than compensate for any required compromise in agility.